Fall 2018

Social Recommendation Systems

Avni Gulati
San Jose State University

SOCIAL RECOMMENDATION SYSTEMS

A Thesis

Presented to

The Faculty of the Department of Computer Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Avni Gulati

December 2018

The Designated Thesis Committee Approves the Thesis Titled

SOCIAL RECOMMENDATION SYSTEMS

by

Avni Gulati

APPROVED FOR THE DEPARTMENT OF COMPUTER ENGINEERING

SAN JOSÉ STATE UNIVERSITY

December 2018

Dr. Magdalini Eirinaki, Ph.D.	Department of Computer Engineering
Dr. David C. Anastasiu, Ph.D.	Department of Computer Engineering
Dr. Katerina Potika, Ph.D.	Department of Computer Science

ABSTRACT

SOCIAL RECOMMENDATION SYSTEMS

by Avni Gulati

In recent years, with the rise of online social networks, personalized recommendations that leverage the aspect of social connections have become a very intriguing domain for researchers. In this work, we explore how influence propagation and the decay in the cascading effect of influence from influential users can be leveraged to generate social graph-based recommendations. Understanding how influence propagates within a social network is itself a challenging problem. In this research, we model the decay in influence propagation in directed graphs, utilizing the structural properties of the social graph to measure the propagated influence beyond one-hop. This social network information from influence propagation is also combined with matrix factorization as a social regularization factor. We then employ this unified framework to form social recommendations, and present our experimental results using real-life datasets. Extensive experimental analysis demonstrate that our proposed methodology outperforms state-of-the-art techniques for generating social recommendations.

ACKNOWLEDGMENTS

I would first like to express my sincerest gratitude to my thesis advisor Dr. Magdalini Eirinaki for her valuable guidance throughout my thesis. Dr. Eirinaki has been an immense source of inspiration for me from the very beginning of this research work. I would like to thank the committee members Dr. Katerina Potika and Dr. David C. Anastasiu for suggesting improvements in this thesis.

I am indebted to my parents who have been a constant pillar of support for me. Last but not the least, I thank my husband Samir, who encouraged me to pursue this thesis. I owe my deepest gratitude to him for his relentless support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Information is a very powerful tool. The information gathered from our friends helps us form opinions and we make decisions based on those opinions. Not only are we influenced by our friends, but friends of friends and even acquaintances can influence our decision making. This idea was first introduced by Mark Granovetter in 1960s when he conducted a study by interviewing people who had recently changed jobs [1]. He found that most of these people had heard about a job opening through an acquaintance and very few got the jobs with the help of personal contacts. He conceptualized this influence from acquaintances as "strength of weak ties" [2], where strong ties are among immediate friends while weak ties act like local bridge between two disconnected group of friends. Granovetter's sociological concepts can be applied to the current online social networks as well. Identifying only a few people who can influence the large majority of the remaining people in the network can be a powerful tool for maximizing social outreach in a social network. This information can be effectively utilized, for example, in the advertising industry by targeting a few popular celebrities on social media to advertise items among the masses, or by government agencies in order to reach the maximum number of people in disease outbreaks and emergencies.

Social recommender systems leverage social relations to improve the rating-based recommendation process [3], based on the assumption that a user's preferences are likely to be similar or influenced by those of his or her friends [4]. Most of existing related work is making the assumption that the users are mostly influenced by their direct neighbors. In this work, we explore the concepts of influential users and influence propagation beyond direct neighbors in the context of social recommender systems.

The focus of this work is three-fold. As a first task, we utilize the structural characteristics of a directed graph to find primary *influentials* and the nodes they *influence*. The objective is to *maximize* influence with the *minimum* number of influencers, and thus

this is defined as a *min-max* problem. We extend previous work [5] that focused on undirected graphs with no cascades, and propose a threshold-bounded influence propagation algorithm that can be applied on directed graphs, taking into account cascading of information in the network. We then propose a two-step recommendation process, in which our proposed algorithm is employed as a pre-processing step to form social graph-based user neighborhoods to be used as input to the recommendation algorithm. Finally, we explore the integration of the social graph input in the recommendation algorithm, by proposing an algorithm that integrates social regularization factors in the matrix factorization process. This step combines the influence propagation step and the recommendation generation process into a unified framework. The proposed high-level methodology is depicted in Fig. 1.

The rest of the paper is organized as follows: we review the related work in the two areas this work touches upon, namely identification of influential users and social recommender systems, in Section 2. After defining our objectives in Section 3, we discuss in detail our proposed methodology and algorithm for identifying influential users in Section 4. We discuss the matrix factorization algorithm and our proposed incorporation of social regularization in Section 5. Finally, in Section 6 we discuss the results of our two-step experimental evaluation of the proposed algorithm, first in the context of influence propagation and social network coverage, and subsequently by indirectly evaluating it as a core component of the social recommendations process. We discuss our plans for future work in Section 7 and finally conclude in Section 8.

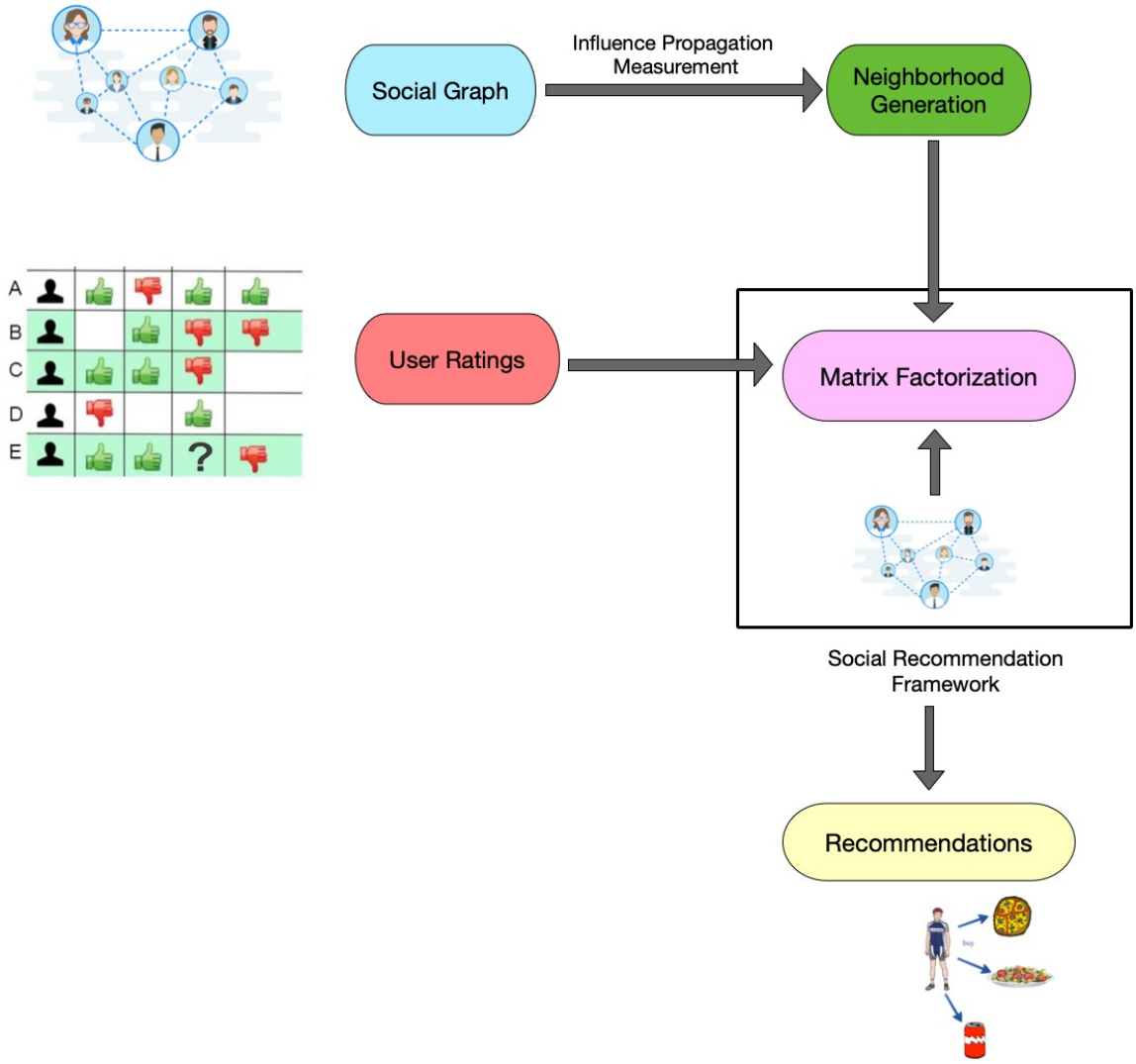

Fig. 1. Unified social recommendation framework.

2 LITERATURE REVIEW

2.1 Identification of Influential Users

The problem of identifying influentials has gained a lot of attention from several research communities as it has applications in viral marketing [6]–[8], disease prevention [9] and propagation [10], politics [11], [12], smart cities [13], software sharing frameworks [14], etc. There are several ways to approach this problem and a few assumptions to be made when one regards this as a graph theoretic one, including whether the graph is directed or not, initialization of the graph, levels of influence propagation and any decay factors associated with it, as well as conditions to activate (i.e. "influence") nodes. This has resulted in a diverse body of research addressing this problem.

Identifying the most influential users of a market was first studied as an algorithmic problem by Domingos and Richardson [15]. They apply data mining techniques to viral marketing, by modeling markets as social networks. They study the spread of influence using probabilistic models of interactions. Every vertex is associated with a value that quantifies how much it can influence other vertices and is used to optimally determine which vertices to choose as influentials. In their empirical study using the EachMovie database, their proposed marketing strategy performs much better than two simple existing strategies.

A more recent work is identifying influentials by calculating correlations between different influence metrics. The authors draw an analogy of Github with Twitter and analyze correlations between follow, mention and retweets in Twitter [16]. Other approaches include using the simulated annealing (SA) algorithm to find top-k influential nodes in a social network. Jiang et al. [17] utilized SA to address the influence overlapping problem. The constraint SA methodology has been used to improve the performance of finding top-k influentials by considering influence loss when certain top nodes fail to function as expected [18]. Two-hop and three-hop problems in social

network are explored in Gong et al. [19] using *particle swarm optimization*, a concept which was first introduced by Eberhart and Kennedy [20].

In this work, we leverage the insights and results of previous work of Eirinaki et al. [5], in which the *NewGreedy* algorithm was introduced to identify influentials in an undirected graph. The premise was that a node is influenced if a number of direct neighbors are "activated" (i.e. become influenced/are influentials) and therefore we assumed that no influence propagation took place. While we follow a similar approach, in this work we focus on directed graphs and assume that influence propagates further than one-hop neighbors. These assumptions pose different challenges and require a novel methodology to be addressed.

Kempe et al. [21] identified the optimization problem of selecting the most influential nodes in a social network as NP-hard. They proposed submodular approaches in a social network diffusion model: *Linear Threshold Model* and *Independent Cascade Model*. In a linear threshold model, a node v randomly chooses a threshold between 0 and 1 and gets activated when the combined effect from its neighbors exceeds the threshold value, only considering neighbors that were active in the previous iteration. In this way, the threshold value dynamically changes with each iteration. In a cascade model, a node u can activate node v with a probability that considers neighbors that have already tried and failed to activate v. In an independent cascade model this probability is independent of neighbors. Motivated by the success of the above models, we also model thresholds dynamically, determined individually for each vertex, as a condition for node activation.

The importance of influence propagation for undirected graphs is well explained by Hangal et al. [22]. They have experimentally shown that the most influential path is more effective compared to the shortest path using Digital Bibliography & Library Project (DBLP) and Twitter datasets by incorporating directed and weighted influence edges in a social graph. They defined a person as influential if he or she has high influence on many

people. They also conceptualized the influence of a node as the sum of all the influence that the node has on others. The most influential path between two nodes was calculated by natural adaptation of Dijkstra's algorithm. In this work, we introduce a decay factor for influence propagation so that it decreases after every hop in the path in a directed network.

2.2 Social Recommender Systems

Social recommender systems have gained a lot of attention in research in an effort to leverage social relationships to improve the recommendation process. This line of work is based on the assumption that users' preferences are influenced more by those of their connected friends, than those of unknown users [8], rooted in the sociology concepts of homophily and social influence [23]. Tang et al. [24] give a narrow definition of social recommendation as "any recommendation with online social relations as an additional input, i.e., augmenting an existing recommendation engine with additional social signals" (a broader definition, not applicable to this work, refers to recommender systems targeting social media domains [25]).

The various proposed approaches can be categorized depending on the type of social relationship (trust, friendship etc.), the type of underlying recommendation algorithm (model-based, memory-based, etc.), and the level of integration of the social information in the recommendation process.

A common approach is to enhance model-based recommender systems with social connections, again most often expressed as trust. This can be done through co-factorization, in which the assumption is that the users share the same preference vector in both the rating and the social spaces (e.g. [26]), ensemble methods, in which the resulting recommendation is derived by the linear combination of two systems (e.g. [27], [28]), or regularization, in which priority is given to the social-based ratings (e.g. [29], [30]).

An alternative line of work involves ways to enhance the memory-based collaborative filtering process by forming the user's neighborhood using similarities deriving from the users' ratings and/or their social relationships, focusing on trust [31]–[37].

Most of the social recommendation research leverages users' similarity for generating recommendations instead of incorporating the social information in the recommendation algorithm itself. Ma et al. [30] introduced social regularization constraints in the recommendation framework. They used the social information to effectively predicted the missing user-item matrix. Experimental results on real world data showed that their algorithm outperformed the traditional matrix factorization methodology.

More recently, Deng et al. [38] introduced a novel deep learning matrix factorization approach to address the issues related to the initialization of latent feature vectors in matrix factorization. They used user's trust of other users belonging to his or her clique as the basis of neighborhood formation. They used the community effect for trust formation among users and integrated it with matrix factorization as a trust regularization factor to predict ratings.

In this work, we follow a similar approach. However, we focus on influence as derived from the social graph connections rather than metadata related to the users. We leverage our proposed influence propagation algorithm and create social graph-based personalized neighborhoods that are subsequently used as input to the recommendation process. Moreover, we combine the social information of the users gained from the influence propagation algorithm to matrix factorization forming a unified social recommendation framework.

3 PROBLEM DEFINITION

The primary objective of this research is to generate effective social recommendations for users incorporating the social information gained from user interactions. This objective can further be subdivided into two broad categories:

- Neighborhood formation using influence propagation.
- Rating prediction using social regularization in matrix factorization.

3.1 Neighborhood Formation Using Influence Propagation

Influence propagation can be defined as a $min - max$ problem: identify the minimum number of most influential nodes (called "seed" nodes), that can influence the maximum number of the remaining nodes (if not the entire network). We then proceed by claiming that this information is leveraged to generate social graph-based recommendations that are more accurate than traditional rating-based ones. Our research objectives can be further divided into the following sub-categories:

- Defining edge-weights for the directed network: We employ structural characteristics of the two nodes forming the edge to determine edge weights. The edge-weights defined are then employed to quantify the influence propagated along vertices in the graphically represented social network.
- Determining condition for influence propagation: We utilize vertex-dependent threshold values in order to determine the condition of whether a node has been influenced by another node or not.
- Ranking nodes: Ranking of nodes is a very critical aspect for determining the top seed users. In this work, several existing methodologies for determining top-k nodes are employed to determine the ones that give optimal results during experimental analysis.
- Neighborhood Generation: We employ the influence propagation approach to generate neighborhood for generating recommendations for users. This results in a

personalized social network-based subset of users, and in turn, in more accurate recommendations.

These goals are discussed in detail in Section 4.

3.2 Rating Prediction Using Social Regularization in Matrix Factorization

As mentioned before, the main goal of this research is to generate effective recommendations, which is achieved by predicting accurate user ratings. This objective is further sub-divided into the following sections:

- Social regularization in matrix factorization: We use low-ranked matrix factorization for user-rating prediction. Neighborhood generated in Section 3.1 is utilized in incorporating social regularization constraints in the traditional matrix factorization technique, producing even more accurate recommendations.
- New similarity metric: We combine similarity among users with the edge-weight defined in the Section 3.1 and employ it with the social regularization in matrix factorization approach. This new similarity integrates influence propagation effect in the recommendation framework.

These objectives are discussed in detail in Section 5.

4 NEIGHBORHOOD FORMATION USING INFLUENCE PROPAGATION

For neighborhood generation using influence propagation, we first focus on how we measure influence propagation in a social network. We define the problem of measurement of influence propagation as follows: Identify the minimum number of seed "influentials", so that there can be the maximum number of "influenced" nodes in the social network. In this section, we describe in detail our approach to this problem.

4.1 Determining Edge Weight in Directed Graphs

A social network can be modeled as a weighted-directed graph $G = (V, E, W)$, where V is the set of all the vertices in the graph (i.e. people), E is the set of edges (i.e. their connections), and W is the set of edge weights. In the directed graph used here, an edge from node u to node v $(u \rightarrow v)$ signifies that user u "follows" v in the social network.

Edge weights are determined in the network by structural characteristics of the two nodes forming the edge. For instance, for the neighboring nodes u and v, the edge-weight $w(u_v)$ represents the *influence* of v on u for this connection, and is defined as:

$$influence(u_v) = importance(v)/outdegree(u) \tag{1}$$

where $importance(v)$ is the in-degree centrality of v. Here, v is "influential" and u is considered an "influenced" node only if $w(u_v) \geq threshold$ value where $threshold$ is a parameter of the algorithm (several strategies for determining its value are discussed in Section 4.2). If this condition holds true, then for generating recommendations v is the seed node and u is considered in v's neighborhood, i.e. $u \in N_G(v)$.

To demonstrate the cascading effect of influence being spread from a single node v, another node p is added that follows u, where $w(p_u)$ is the respective edge-weight. This can be graphically represented as $p \rightarrow u \rightarrow v$. Node p is at two degrees of separation from

v. Hence, if u is influenced by v, then we claim that node p is also influenced by v (and therefore $p \in N_G(v)$) if

$$w(p_u) * hopping\ factor \geq threshold \tag{2}$$

Here, we have introduced a $hopping\ factor$ for capturing the essence of decay in information or influence with increasing hops which has been defined as:

$$hopping\ factor = 1 + hop * decay \tag{3}$$

where hop is an integer value 0 for immediate neighbors (i.e. adjacent nodes) and it increments by 1 for each subsequent hop, and $decay$ is a constant equal to 0.1[1].

4.2 Determining Condition for Influence Propagation

In the previous section, $threshold$ was introduced to determine if u is influenced by v. Here, three threshold conditions are proposed for finding if a node is influenced:

- Condition 1: No threshold (*NoThr*): The first condition is considered for two-hops in the graph considering no threshold for influence propagation.
- Condition 2: Average threshold (*AvgThr*): The second condition takes threshold as the average of edge-weights of the entire network. This would mean that the threshold is constant for all the nodes, depending on the characteristics of the network as a whole.
- Condition 3: Edge-weight dependent threshold (*EWThr*): The third condition determines the threshold by taking average of edge-weights of all the outgoing-edges from the node in the graphically represented social network. Thus, this threshold condition is vertex-specific but constant for every node.

[1]Value set after experimentation

4.3 Methodologies for Ranking Nodes

This research intends to find the optimal ranking strategy for nodes that initiate the process of influence propagation in social graph G (mentioned in Section 4.1). There are several ways proposed in the literature to identify important nodes in a graph. Most of those approaches use a graph-based metric such as centrality (degree, eigenvector, etc.) or PageRank to generate an initial ranking of nodes. Here, three methodologies for ranking of nodes are employed:

- PageRank (*PR*): Page et al. [39] introduced PageRank for ranking importance of web pages. This is a very popular methodology used in social networks for ranking nodes.
- Out-degree centrality (*Outdeg*): Out-degree centrality measure is utilized here for initializing nodes to measure influence outreach in the social graph.
- Upper-bound PageRank (*UB-PR*): Liu et al. [40] used PageRank to find authority of nodes. They evaluated upper bounds of PageRank to find top authorities and introduced an efficient way to rank nodes. Here, this upper bound ranking is employed as one of the initial ranking criterion for nodes in the social network.

The process of identifying influential nodes is performed in sequential order starting with the top ranked nodes. These nodes are obtained from one of the ranking methodologies stated above. For each influential node considered in this list, influenced nodes are identified by following the methodology mentioned in Section 4.1. All the nodes that have been influenced once are removed from consideration of getting influenced by the successive nodes.

4.4 Algorithm

We follow the notation introduced previously and model the social network as a weighted-directed graph $G = (V, E, W)$, an edge from node u to node v $(u \rightarrow v)$ signifies that user u "follows" v in the social network, and the edge weight $w(u_v)$ represents the influence of v on u. Our objective is to identify the influential nodes $v \in M, M \subset V$ that

influence the remaining nodes $u \in D, D \subset V$, $D \cap M = \emptyset$ such that $|M|$ is minimized and $|D|$ is maximized. In this section, we introduce the *Threshold-Bounded Influence Propagation in Digraph (TB-IP)* that takes as input a graph G and outputs sets M and D. The algorithm accepts as parameters the following: a) a threshold thr that is defined according to the selected threshold condition (as described in Section 4.2), b) a maximum number of "hops" (i.e. the maximum allowed depth of influence propagation), c) a decay factor for the influence propagation, and d) a ranking strategy for initializing the nodes (as described in Section 4.3).

The algorithm begins by sorting all nodes in descending order based on their assigned rank $r(v)$. Then, beginning with the highest ranked node, it examines whether its direct connections should be added to its neighborhood and therefore be considered influenced or not. This is determined by examining whether the edge-weight $w(u_v)$ of a connected node u is above the set threshold thr or not. If at least one connected node qualifies, node v is being added to the "influencer" set M and the qualifying nodes are being added to the "influenced" set D. If the algorithm is set to examine nodes that are indirectly connected to v (depth is defined by the *maxhop* parameter), each of the nodes that were added to $N_G(v)$ in the previous step are used to find their directly connected nodes. However, in this case, the respective edge-weights are updated by the $hopping\ factor$, as defined in Section 4.1, before being evaluated against the threshold thr. When a node satisfies the threshold condition and has not been previously added to the "influenced" set D, then it is being added to both this set, and the neighborhood of v, $N_G(v)$. The algorithm stops this loop when either the maximum depth (i.e. number of hops) has been reached, or no nodes are qualifying as "influenced" in the current level. This process is being repeated for each of the nodes, as selected from the ranked list, and as long as they have not already been added in the "influenced" set D. The above process is described in detail in Algorithm 1.

Algorithm 1 Threshold-Bounded Influence Propagation in Digraph (TB-IP)

Require: A weighted and directed social network $G = (V, E, W)$
Ensure: Influenced Vertices $D \subset V$ and Influential Vertices $M \subset V$

Initialize: thr, $maxhop$, $visit = 0$, $hop = 0$, $decay = 0.1$, $D = \emptyset$, $M = \emptyset$
$\forall v \in V$, $r(v)$ = compute_rank(v)
Sort $\forall v \in r(v)$ in non-increasing order
for each $v \in S$ following order **do**
    if $w(u_v) > thr$ and $\exists\ u \in V \setminus D$ **then**
        $N_G(v) = N_G(v) \cup u$
        $D = D \cup u$
        $M = M \cup v$
        $visit = 1$
    while $hop \leq maxhop$ and $visit = 1$ **do**
        $hop = hop + 1$
        $visit = 0$
        for each $u \in N_G(v)$ **do**
            $w(p_u) = w(p_u) * (1 + hop * decay)$
            if $w(p_u) > thr$ and $\exists\ p \in V \setminus D$ **then**
                $N_G(v) = N_G(v) \cup p$
                $D = D \cup p$
                $visit = 1$
output D, M

As we can see, that we go over all the nodes in the social graph as any node can be a potential $influential$. Once a node has been "influenced", we donot revisit it. This continues till maximum hops. Hence, the running time of TB-IP algorithm is $O(n \times m^{maxhop})$. For experimental purposes, we limit the $maxhop$ to 2 or 3 in this research.

5 MATRIX FACTORIZATION

A lot of techniques have been used in the past to generate effective recommendations for users. One of the most popular methodology is collaborative filtering. This research employs matrix factorization for collaborative filtering to generate recommendations for users. Matrix factorization uses dimensionality reduction to find the latent features of correlated user-item interactions. Let us consider an example of products that are reviewed by customers using a web application. There are thousands of users who explicitly review millions of products. We can represent these reviewed products as the user's feature vectors, and hence, we can leverage the correlation between the reviewed products and reduce the dimensions of user's feature vector [41].

Mathematically, the user-item matrix (R) can be decomposed into two matrices, namely user (P) and item (Q). On multiplying user and item matrices, we can regenerate the utility matrix. Considering the dimensions of R as $m \times n$, P and Q can be represented in smaller dimensions $m \times k$ and $n \times k$ respectively. The low-ranked matrix factorization approach would generate the user-item matrix as

$$\underset{m\times n}{\mathrm{R}} \approx \underset{m\times k}{\mathrm{P}^{\intercal}} \times \underset{n\times k}{\mathrm{Q}} \tag{4}$$

Here, $k < min(m,n)$. R is a sparse matrix because users tend to rate very less percentage of products that are available. Matrix factorization of R diagrammatically presented in Fig. 2.

Singular value decomposition is generally used to factorize the observed ratings from the utility matrix to obtain the latent factors by minimizing the following term:

$$\min_{P,Q} \frac{1}{2} \sum_{u=1}^{m} \sum_{i=1}^{n} I_{ui}(R_{ui} - P_u^T \cdot Q_i)^2 + \lambda_1(\|P\|_F^2) + \lambda_2(\|Q\|_F^2), \tag{5}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are the regularization factors added to avoid overfitting. Moreover, I_{ui} is an indicator function which is 1 when user u has rated item i and it is

equal to 0 if the user has not rated the item. $\|.\|_F^2$ denotes Frobenius norm. Several optimization approaches can be used to find local minimum of Equation 5, such as alternating least squares or stochastic gradient descent.

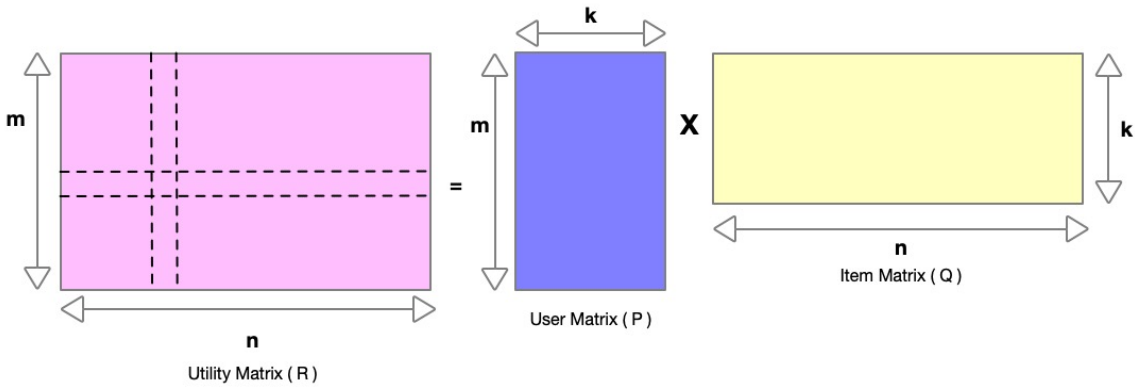

Fig. 2. Pictorial representation of matrix factorization.

5.1 Social Regularization in Matrix Factorization

Ma et al. [30] introduced a social factor in the above mentioned matrix factorization framework. This social factor leverages social network information gathered for a user and incorporates this social aspect to generate better recommendations. They introduced two approaches to integrate this social information in matrix factorization. The first approach uses average-based regularization, which assumes that every user's taste is close to the average taste of the user's friends. But this approach does not take into consideration the friends of a user who have varied tastes which could lead to inaccurate results. The second methodology incorporates individual-based regularization that solves the drawback of the previous approach.

In this research, we leverage and extend the second approach used by Ma et al. [30], i.e. individual-based regularization approach. They minimize the objective function mentioned in Equation 5 as well as the social regularization term as follows:

$$\begin{aligned}\min_{P,Q} \mathscr{L}_1(R,P,Q) = \frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} I_{ui}(R_{ui} - P_u^T \cdot Q_i) + \\ \frac{\beta}{2}\sum_{v\in\mathscr{F}^{+(\mathrm{u})}} Sim(u,v)(\|P_u - P_v\|^2) + \\ \lambda_1(\|P\|_F^2) + \lambda_2(\|Q\|_F^2),\end{aligned} \tag{6}$$

where $\beta > 0$, $\mathscr{F}^{+(\mathrm{u})}$ denote user u's directly connected out-link friends. Following the similar convention, $\mathscr{F}^{-(\mathrm{u})}$ would denote user u's direct in-link friends. $Sim(u,v) \in [-1,1]$ is the similarity between user u and user v. This is defined in the subsequent paragraphs.

According to Ma et al. [30], a local minimum of Equation 6 can be obtained by applying stochastic gradient descent on the P_u and Q_i feature vectors. Hence, they evaluate partial derivatives on both the feature vectors as shown in Equation 7 and 8

$$\begin{aligned}\frac{\partial \mathscr{L}_1}{\partial P_u} = \sum_{i=1}^{n} I_{ui}(P_u^T \cdot Q_i - R_{ui})Q_i + \lambda_1 P_u \\ \beta \sum_{v\in\mathscr{F}^{+(\mathrm{u})}} Sim(u,v)(P_u - P_f) + \\ \beta \sum_{p\in\mathscr{F}^{-(\mathrm{u})}} Sim(u,p)(P_u - P_p)\end{aligned} \tag{7}$$

$$\frac{\partial \mathscr{L}_1}{\partial Q_i} = \sum_{u=1}^{m} I_{ui}(P_u^T \cdot Q_i - R_{ui})P_u + \lambda_2 Q_i \tag{8}$$

Note that we have used a similar user convention used in Section 4 in which user p follows user u and user u follows user v.

The similarity used in Equation 6, 7 and 8 is defined as follows:

$$Sim(u,v) = \frac{1}{2}(PCC(u,v)+1) \tag{9}$$

where, *PCC* is the Pearson correlation coefficient [42] for user u and user v and it is formulated as:

$$PCC(u,v) = \frac{\sum_{i \in I_u \bigcap I_v} (R_{ui} - \bar{R}_u)(R_{vi} - \bar{R}_v)}{\sqrt{\sum_{i \in I_u \bigcap I_v} (R_{ui} - \bar{R}_u)^2} \sqrt{\sum_{i \in I_u \bigcap I_v} (R_{vi} - \bar{R}_v)^2}}, \tag{10}$$

Here, i is the subset of total items user u and user v have both rated. R_ui is the rating given by user u to item i. $\bar{R}_u$ and $\bar{R}_v$ are the average ratings of users u or v respectively and R_{ui}, R_{vi} are the ratings of users u and v respectively for item i. It is a very popular similarity metric. A higher numerical value of $PCC(u,v)$ means that u and v are more similar to each other and a smaller value of $PCC(u,v)$ implies that u and v have dissimilar choices.

5.2 Influence Propagation in Social Regularization

In this work, we extend the above mentioned process (Section 5.1) by employing the TB-IP algorithm discussed in Section 4.4 in two ways, as shown in Fig. 1. First, we apply TB-IP on the social graph to define the neighborhood $N_G^{(u)}$ for each user u, as a pre-processing step.

We propose to incorporate the social regularization in our matrix factorization objective function using the Equation 5 for N_G. In addition, contrary to Ma et al. [30], who have formulated similarity among users using only the conventional PCC, we utilize edge-weights calculated from Equation 1 as social relation weights in the similarity function (Equation 9). Hence, we propose a new similarity metric which is the mean of

the social edge weights and PCC between user u and user v as follows:

$$SocSim(u,v) = \frac{1}{2}(PCC(u,v) + w(u_v)) \tag{11}$$

This new similarity, *SocSim*, captures the social information from the influence propagation using TB-IP algorithm 4.4 in matrix factorization as a part of its regularization term. Now, we can redefine Equations 6, 7 and 8 utilizing our proposed similarity metric,

$$\begin{aligned}\min_{P,Q} \mathscr{L}_2(R,P,Q) = \frac{1}{2}\sum_{u=1}^{m}\sum_{i=1}^{n} I_{ui}(R_{ui} - P_u^T \cdot Q_i) + \\ \frac{\beta}{2}\sum_{v \in N_G^{+(u)}} SocSim(u,v)(\|P_u - P_v\|^2) + \\ \lambda_1(\|P\|_F^2) + \lambda_2(\|Q\|_F^2),\end{aligned} \tag{12}$$

$$\begin{aligned}\frac{\partial \mathscr{L}_2}{\partial P_u} = \sum_{i=1}^{n} I_{ui}(P_u^T \cdot Q_i - R_{ui})Q_i + \lambda_1 P_u \\ \beta \sum_{v \in N_G^{+(u)}} SocSim(u,v)(P_u - P_v) + \\ \beta \sum_{p \in N_G^{-(u)}} SocSim(u,p)(P_u - P_p)\end{aligned} \tag{13}$$

$$\frac{\partial \mathscr{L}_2}{\partial Q_i} = \sum_{u=1}^{m} I_{ui}(P_u^T \cdot Q_i - R_{ui})P_u + \lambda_2 Q_i \tag{14}$$

where $\beta > 0$, ${N_G}^{+(u)}$ denote u's out-link connections belonging to its neighborhood, ${N_G}^{-(u)}$ denote u's in-link connections belonging to its neighborhood, $SocSim(u,v)$ is defined in Equation 11.

6 EXPERIMENTAL EVALUATION

Our evaluation focuses on three main objectives: a) evaluate the proposed TB-IP algorithm, b) evaluate the effect of the social graph-based neighborhood as input to the recommendation process, and c) evaluate the integrated social recommendation framework with TB-IP neighborhood social regularization and proposed *SocSim* metric.

6.1 TB-IP Algorithm

In order to evaluate our proposed algorithm, we employed three real-world social network datasets that are being broadly used in similar experimental setups, namely Epinions [43], Astro Physics [44] and Yelp[2]. Epinions is an online social network that has product reviews by customers. It is a directed trust network in which customers express their trust or distrust to other reviewers in the network. AstroPhysics is a scientific collaborative network among authors who have published their papers in the AstroPhysics domain. It is an undirected network which is changed into a directed graph for experimentation purposes by adding one edge for each direction. Yelp is a business review and recommendation service in which users primarily review restaurants and rate them. This dataset is taken from the 2018 Yelp Challenge. It is a directed network in which users can be ”friends” with each other. Since the Yelp social subset (i.e. the friends' network) is very sparse, we only considered users who had rated at least one business in the city of Las Vegas, one of the most dense social graph subsets. For these users, we considered only those connections who rate restaurants and reside in the same area. We will refer to this as the Yelp Las Vegas dataset. The network properties of the three datasets are given in Table 1.

As mentioned earlier, this research extends the *NewGreedy* algorithm proposed by Eirinaki et al. [5]. The *EWThr* threshold approach is equivalent to finding activated nodes evaluated by the *NewGreedy* approach, in which the threshold in an undirected graph is

[2]https://www.yelp.com/dataset/challenge

degree-dependent for every node. In that work, the threshold changed dynamically by removing the nodes in the graph that were already activated. In the methodology proposed in this paper, the *EWThr* condition for each node in a directed graph does not change dynamically but remains constant (yet can be vertex-specific). However, since the edge-weights determined are degree-dependent here, an analogy can be formed between this methodology with the *NewGreedy* algorithm.

Table 1
Dataset Characteristics

Dataset	**Astro Physics**	**Epinions**	**Yelp (Las Vegas)**
# Nodes	18,772	75,879	247,111
# Edges	198,050	508,837	5,340,568
Avg. Degree	21.100	6.706	21.612
Type	Undirected	Directed	Directed

In order to compare and draw conclusions easily, we replicated the setup used in the experimental evaluation of [5]. Therefore, for each of the three datasets, we generated subgraphs by randomly picking nodes as seed nodes and adding their neighbors so that the graph has more than 100 vertices but fewer than 600 vertices. These subgraphs were used as input to our experiments. Moreover, we set random seed equal to 1 for generating consistent results every time the algorithm runs.

Our objective is to compare the various ranking methodologies and threshold strategies in terms of the defined min-max problem. In other words, a setup is preferable when we are able to influence the maximum percentage of nodes with the minimum number of influencers. The results for the three datasets, namely AstroPhysics, Epinions, and Yelp, are shown in Figs. 3, 4 and 5 respectively. In each figure, the first column shows the percentage of influenced nodes for different sizes of graphs (so higher is better), whereas the second column shows the respective number of influencers needed for the respective coverage (therefore lower is better).

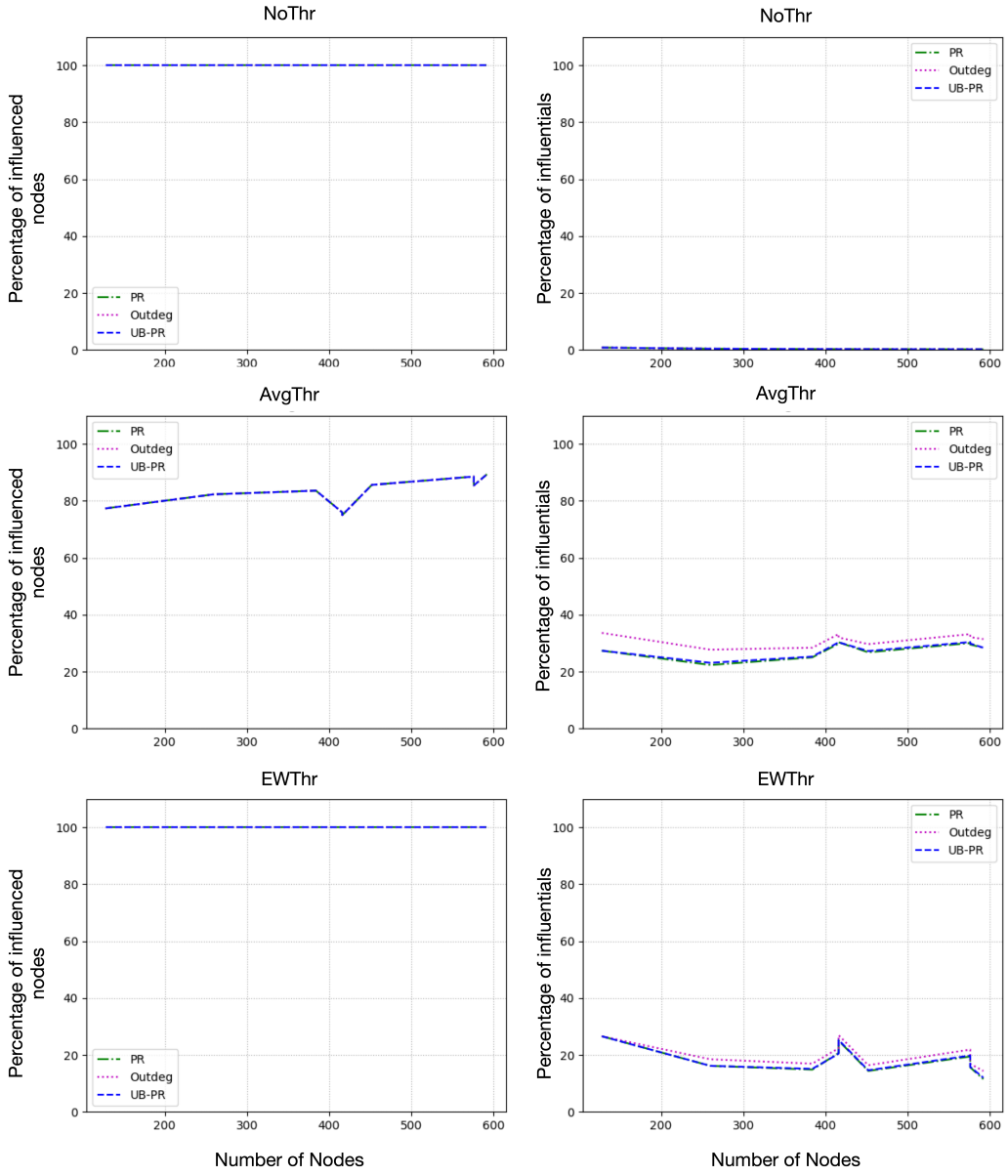

Fig. 3. Experimental analysis of TB-IP algorithm on Astro Physics dataset.

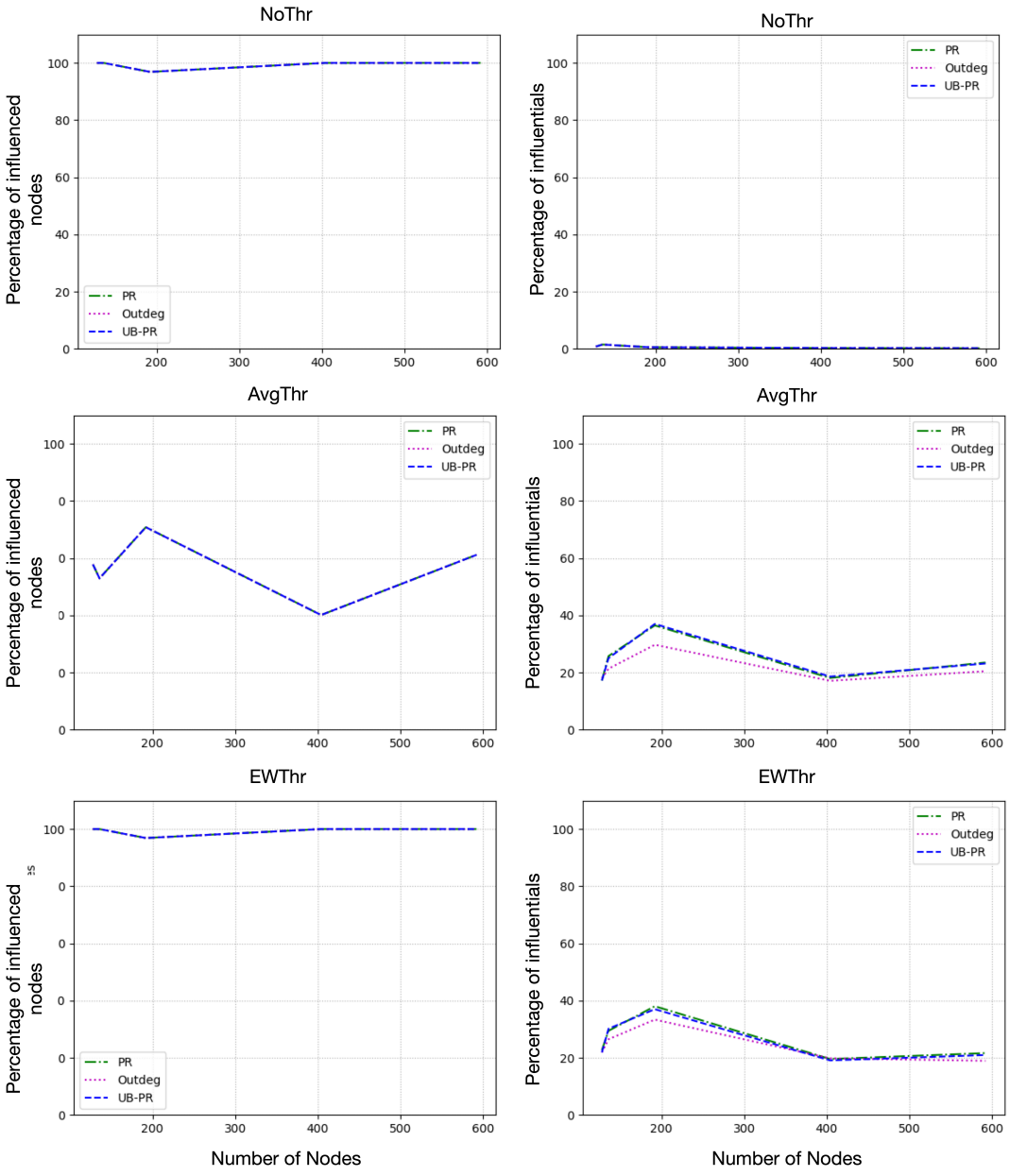

Fig. 4. Experimental analysis of TB-IP algorithm on Epinions dataset.

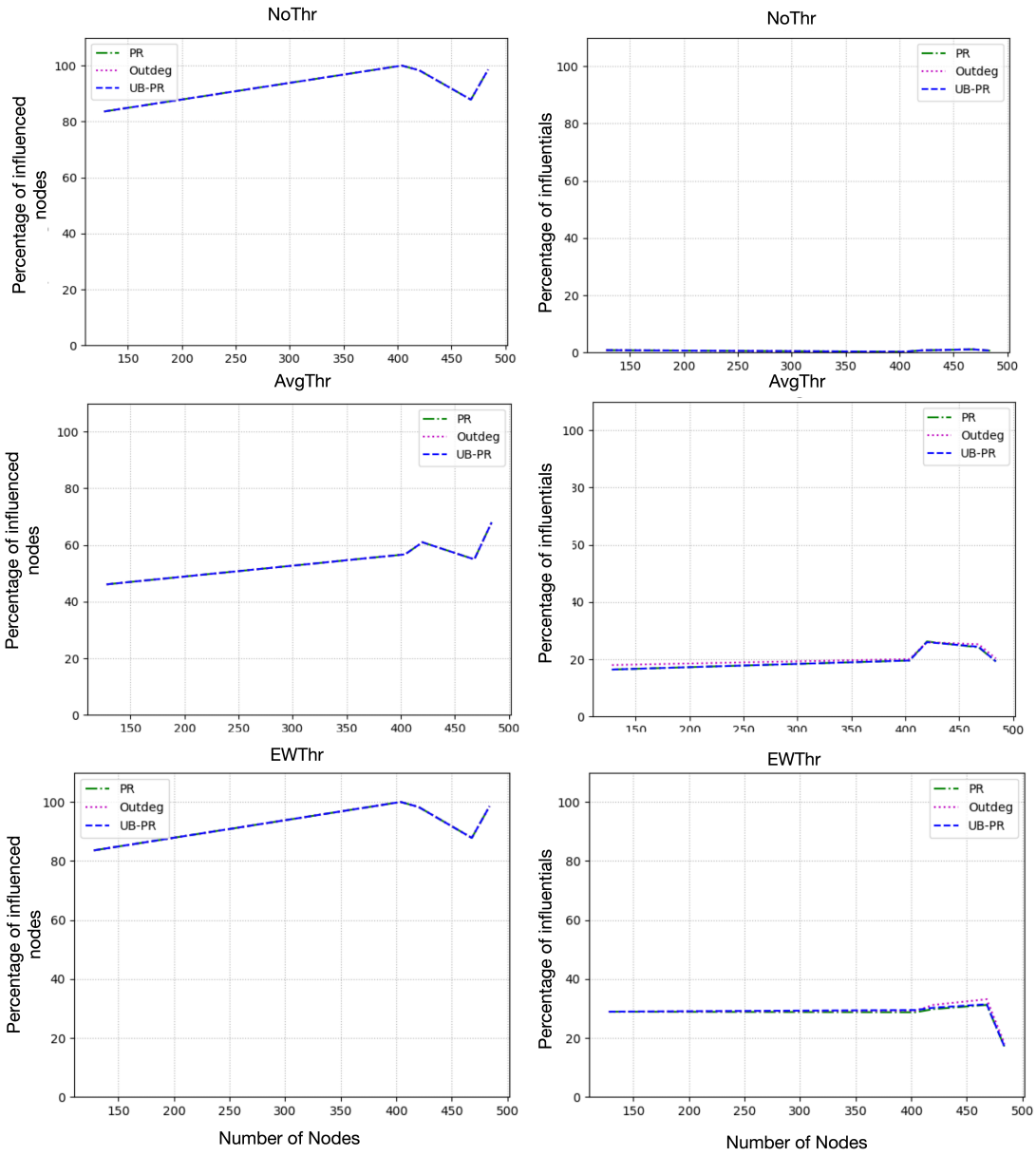

Fig. 5. Experimental analysis of TB-IP algorithm on Yelp dataset.

The first row in these figures, corresponding to the no threshold condition (*NoThr*) is regarded as our naive baseline. This approach assumes that influence propagates for all neighboring nodes of any node in the graph, and up to two degrees of separation (regardless of this node's initial influence score). This approach is expected to achieve maximum coverage in very few steps, especially for small and highly connected graphs. However, this is a naive approach as it assumes that all nodes spread influence equally, an assumption that does not usually hold in social networks. However, we include here the results and use them as a baseline in order to assess the remaining two conditions, namely average threshold (*AvgThr*) and edge-weight dependent threshold (*EWThr*). These conditions are experimentally analyzed and are depicted in the figure in second and third row, respectively. Moreover, the three ranking methodologies mentioned in Section 4.3 are depicted in different colors and different dotted lines. Sequentially, green is for *PR*, pink for *Outdeg* and blue represents *UB-PR* ranking.

Indeed, we observe that in all networks the *NoThr* condition achieves very high (or maximum) network coverage with just a few influencer nodes. As implied previously, this was expected by design. As the algorithm starts with the highest ranked nodes and ranking is employing (outdegree or PageRank) centrality, the most central nodes cover a huge percentage of the graph in two hops, especially for such small subgraphs. However, this is not a realistic condition. We therefore focus on the performance of the other two conditions, which set thresholds on when influence propagates.

Moreover, we observe that the coverage achieved by the *EWThr* approach is similar to the baseline one, influencing 80% to 100% of the nodes, while the *AvgThr* approach does not manage to cover the network in its entirety, influencing between 40% and 90% of all of the nodes. This is because some nodes in these real-world networks have numerically lower in-degree values while other nodes have numerically higher out-degrees making the edges-weights so small that even an immediate neighbor is unable to influence the

adjacent node. On the other hand, the *EWThr* threshold condition depends on structural characteristics of every node, and this might be the reason for its better performance.

Looking more closely at the ranking methodologies, we observe that all are comparable in terms of number of influencers, with *Outdeg* performing slightly better for the AstroPhysics and Yelp subgraphs, and *PR* performing slightly better for the Epinions subgraph. However the differences are minimal (possibly due to the size of the subgraphs) and therefore we cannot draw definite conclusions.

Finally, looking at the performance of the *NewGreedy* algorithm on the AstroPhysics dataset, which is the only common one (the graph was considered undirectional in [5] [3]), we observe comparable results.

Note that in the implementation of the influence propagation methodology proposed in this paper, the graph is reversed for finding the influence of a node so that incoming edges become outgoing edges and vice versa. This is done to capture influence (incoming edge) of a node. By reversing the incoming node, the outgoing reach of a particular node can be determined in the entire network. This can be understood by taking an example from the Twitter social network, in which a person may follow Obama but Obama does not follow him or her. In the graph, this will be represented as an incoming edge toward Obama. However, for the purpose of our algorithm, and in order to find the influence of Obama (and, by extension, of all users) over the entire network, the edge directions should be reversed.

Moreover, the edge-weights are normalized by taking the logarithm of their values. This generates a uniform distribution of edge-weights. The new normalized values of edge-weights are negative, hence to decrease the influence of edge-weights with increasing cascade, the $hopping\ factor$ should be greater than 1 (Equation 3), making edge-weights of subsequent hops smaller in numerical value.

[3] Results are shown in [5] - Fig. 5.

6.2 Neighborhood Formation

In this experiment we evaluate different neighborhood formation approaches that leverage influence propagation in the social graph to enhance the accuracy of a recommender system. Using the proposed TB-IP algorithm described in Section 4.4, we generated neighborhoods of the "influential" users in the Yelp social network, derived from the Yelp Las Vegas dataset [4].

In this experimental analysis, we use the traditional matrix factorization approach, i.e. without using any social regularization for generating recommendations. We designed three social graph-derived datasets to be used as input to the recommender system, by considering the three different strategies of *threshold* selection. For our baselines we generated datasets starting with the same seed users as the social graph-based ones, and adding equal number of additional users. The difference is that the latter were randomly selected (i.e. the social graph elements were not taken into consideration). In total, we created seven different datasets to be used as input to the recommendation process, resulting in seven different experiments. We should point out that, since the actual users included in each dataset are different, the number of respective ratings and businesses is also different, as shown in Table 2. Additionally, for *TwoD NoThr* dataset, we considered only those seed nodes that influenced minimum 5 unique nodes. For both *ThreeD AvgThr* and *TwoD EWThr*, the top-ranked node itself influenced a large portion of other nodes in the network.

- Dataset 1- Two hops with no threshold (*TwoD NoThr*): In this experiment we considered two hops in the network and no *threshold* for influence propagation for forming neighborhood in the algorithm above. For this, we selected 2982 seed users and expanded it by adding users influenced by these seed users, forming their neighborhood.

[4]We used the following attributes: *user id*, $friends\ ids$, *business id*, and *ratings*.

Table 2
Yelp Dataset Characteristics for Neighborhood Formation Experiments

Dataset	# Users	# Businesses	# Ratings
TwoD NoThr	146,938	25,611	859,310
BaseD1	146,938	25,028	661,414
ThreeD AvgThr	182,464	25,799	953,237
BaseD2	182,464	25,703	815,350
TwoD EWThr	96,900	24,632	733,408
BaseD3	96,900	23,245	432,714
BaseAll	247,111	26,304	1,104,768

- Dataset 2- Baseline Two (*BaseD1*): This experiment is baseline for Dataset 1 in which we have expanded the original dataset of 2982 seed users by randomly selecting other users.
- Dataset 3- Three hops with average threshold (*ThreeD AvgThr*): In this experiment we considered three hops in the network and the average of all edge-weights as *threshold* condition for considering a node as influenced. For this, we took top-ranked seed user and expanded it by adding users influenced by this user.
- Dataset 4- Baseline Three (*BaseD2*): This experiment is baseline for Dataset 3 in which we have expanded the original dataset of the seed user by randomly selecting other users to make the total number of users equal.
- Dataset 5- Two hops with edge-weight dependent threshold for every node (*TwoD EWThr*): We set the *threshold* for a node u to be influenced to be equal to the average edge-weight $w(u_v)$ of outgoing edges and consider influence propagation for up to 2 hops. Thus, this *threshold* is vertex-dependent. To form the input dataset, we begin by the top-ranked user and expand the neighborhood by adding users influenced by this seed user.
- Dataset 6- Baseline One (*BaseD3*): We create a baseline dataset for Dataset 5 by randomly selecting an equal number of users (the seed user is included in this set).

- Dataset 7- Baseline All (*BaseAll*): In this experiment, we used the entire dataset including all Yelp users who have rated at least one business in Las Vegas.

We used the 7 datasets as input to the matrix factorization algorithm[5] and performed 10-cross-fold-validation for all possible combinations of the following parameters: rank $(f) = \{5, 10, 20\}$, regularization $(\lambda) = \{0.1, 0.05, 0.01, 0.5, 0.2\}$, and number of iterations, or max iter = $\{10, 20\}$ to evaluate the results.

For this purpose we employed the popular error-based metrics root-mean-square error (RMSE) and mean absolute error (MAE). Assuming that the recommender system generates predicted ratings $\hat{R}_{ui}$ for a test set T of user-item pairs (u,i) for which the true ratings R_{ui} are known. RMSE between the predicted and actual ratings is given by:

$$RMSE = \sqrt{\frac{1}{n} \sum_{(u,i) \in T} (\hat{R}_{ui} - R_{ui})^2} \tag{15}$$

where n is the size of set T. MAE is a simpler alternative, given by:

$$MAE = \frac{1}{n} \sum_{(u,i) \in T} |\hat{R}_{ui} - R_{ui}| \tag{16}$$

The best (i.e. lowest) RMSE and MAE for each Dataset/Experiment are reported in Table 3[6]. We observe that the social graph-based dataset/experiment combos outperform the respective baselines. Moreover, we observe that the vertex-specific threshold strategy (*EWThr*) outperforms the other two. This confirms our assumption that, while the *NoThr* strategy seemed to perform better in terms of coverage/number of influentials in our previous experiments with the small subgraphs, it is a naive approach that will not reflect real-life relationships depicted in large social networks like the one we used here.

Therefore, we verify our initial intuition that the recommendation process greatly benefits when enhanced with social graph data. In addition to that, we observe that the

[5]We used the alternating least squares matrix factorization in pySpark MLlib library.
[6]The best settings were f=5 and λ=0.5 for RMSE, and f=5 and λ=0.2 for MAE.

dynamic threshold strategy is the dominating one in this implicit evaluation of the algorithm as well.

Table 3
RMSE and MAE for Different Neighborhood Formation Setup

Experiment	RMSE	MAE
TwoD NoThr	1.33	1.04
BaseD1	1.39	1.11
ThreeD AvgThr	1.34	1.05
BaseD2	1.39	1.10
TwoD EWThr	**1.29**	**1.01**
BaseD3	1.40	1.12
BaseAll	1.38	1.09

6.3 Matrix Factorization with Social Regularization

In the previous Section 6.2, we used the Yelp dataset to generate accurate recommendations. Now, we use the best resulting neighborhood-generation strategy, i.e. $EWThr$ with Epinions dataset to evaluate the recommendations combining social regularization aspect defined in Section 3.2.

Tables 4 and 5 depict user-item rating and trust characteristics of Epinions dataset respectively. We perform data analysis on entire Epinions dataset, the dataset characteristics are mentioned in Table 6.

Table 4
Epinions User-Item-Rating Dataset Characteristics

Dataset	Epinions
# Users	22,164
# Edges	355,217
# Reviews	922,267

- Dataset 1 (*BaseAll*): This dataset has the entire Epinions user-item-rating data,
- Dataset 2 (*TwoD EWThr*): This has the rating data only for users obtained after running TB-IP algorithm with $EWThr$ threshold condition. Here, we utilize $Outdeg$ ranking methodology for initial ranking of nodes.

Table 5
Epinions Trust Dataset Characteristics

Dataset	Epinions
# Users	18,089
# Edges	355,217
# Reviews	597,579
Avg. Degree	19.6372
Type	Directed

Table 6
Epinions Dataset Characteristics Used in Experimental Analysis

Dataset	# Users	# Items	# Ratings
BaseAll	22,164	296,277	922,267
TwoD EWThr	7,615	222,281	539,963

We consider two baseline methodologies to evaluate our proposed social recommendation framework. First baseline evaluates the traditional matrix factorization (*Basic MF*) explained in Equation 5. As the second baseline we consider the social regularization in matrix factorization (*SoReg*) developed by Ma et al. [30] which was elaborated in Equations 12, 13, 14. The baseline is their proposed similarity function mentioned in Equation 9 (*Sim*). We compare this against our method, employing the new similarity metric function (*SocSim*) that we defined in Equation 11.

We have performed experiments with several different setup to evaluate their accuracy with real world dataset.

- Experiment 1 (*SimpleMF*): Here, we took *BaseAll* dataset from Table 6 and implemented *Basic MF*. This approach is the first baseline for our experimental analysis.
- Experiment 2 (*TB-IP SimpleMF*): In this experiment, we considered dataset *TwoD EWThr* and generated recommendations using *Basic MF*.
- Experiment 3 (*Simple SoReg*): Here, we evaluated *SoReg* methodology with *Sim* for *BaseAll* dataset. Hence, this is our second baseline algorithm.

- Experiment 4 (*TB-IP SoReg*): In this experiment, we considered dataset *TwoD EWThr* and generated recommendations using *SoReg* technique using *Sim* similarity metric
- Experiment 5 (*SocSim Simple SoReg*): Here, we evaluated *SoReg* methodology with *Sim* similarity metric for *BaseAll* dataset. Hence, this is our second baseline algorithm.
- Experiment 6 (*SocSim TB-IP SoReg*): In this experiment, we considered dataset *TwoD EWThr* and generated recommendations using *SoReg* technique with *SocSim* similarity metric.

This experimental setup is depicted in Fig. 6. The highlighted portions in this figure are the methodologies proposed in this work. They are examined with the baseline recommendation approaches which are depicted in gray-scale.

We performed 5-fold cross validation for all the experiments performed in this section.[7] We used all the possible combination of following parameters for matrix factorization: $f = \{10,20\}$, $\lambda = \{0.1,0.01,0.001\}$, $max\ iter = \{10,20\}$. For the *SoReg* technique we utilized another metric, called *social regularization* $(\beta) = \{0.1,0.01\}$. We employ RMSE and MAE evaluation metric as before to evaluate our recommendation models. The results are presented with the corresponding experiment performed in Table 7 and Table 8 for RMSE and MAE respectively.

Table 7
RMSE and Optimal Parameter Settings for Different Experiments Performed

Experiment	**RMSE**	**f**	λ	β	**max iter**
SimpleMF	1.063	20	0.001	0	10
TB-IP SimpleMF	1.046	20	0.001	0	10
Simple SoReg	1.046	20	0.01	0.1	20
TB-IP SoReg	**1.030**	20	0.01	0.1	20
SocSim Simple SoReg	1.061	20	0.01	0.1	10
SocSim TB-IP SoReg	1.043	20	0.01	0.1	10

[7]We used ReQ python library from https://github.com/Coder-Yu/RecQ to perform our experiments.

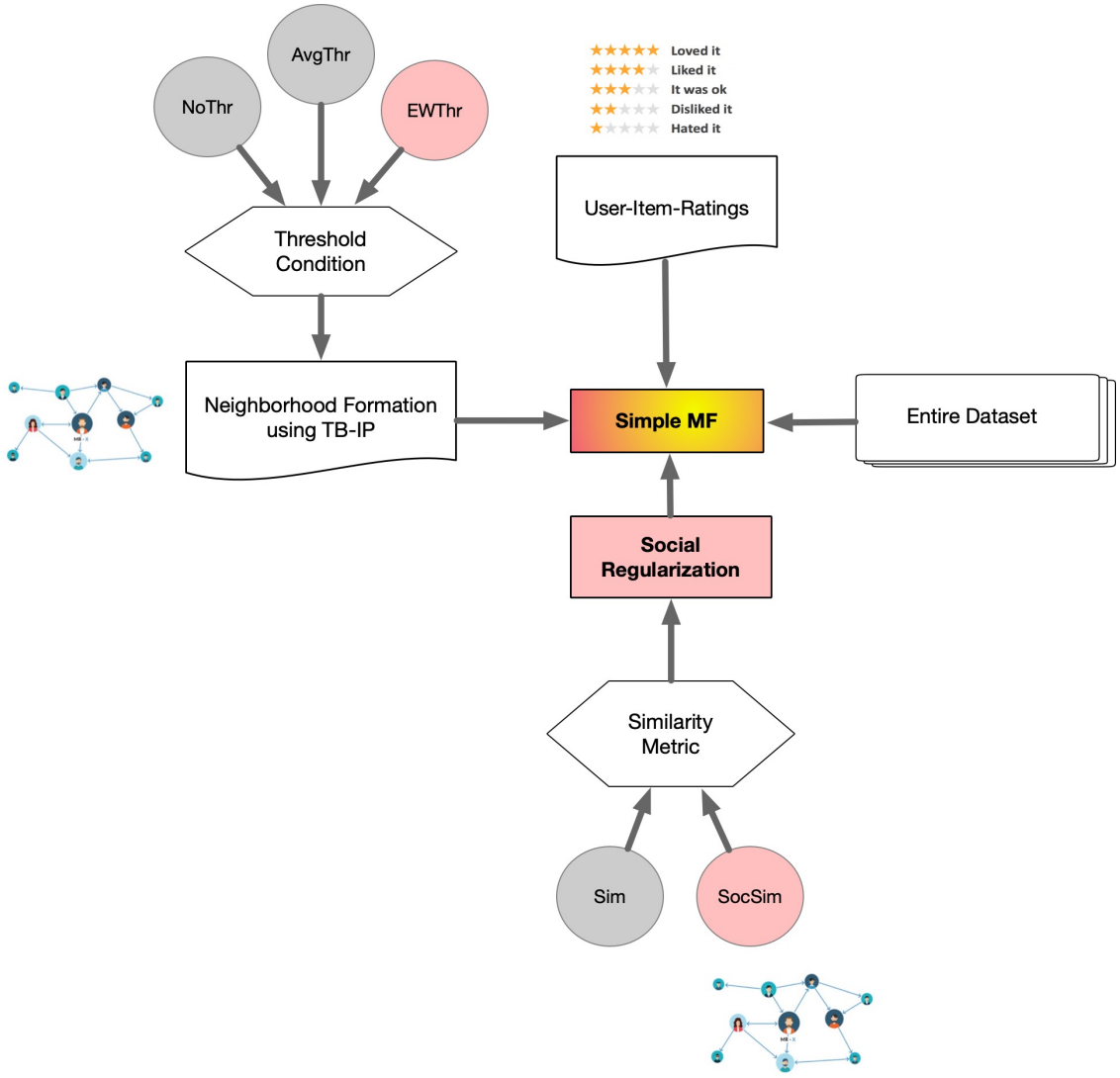

Fig. 6. Experimental setup.

Table 8
MAE and Optimal Parameter Settings for Different Experiments Performed

Experiment	MAE	f	λ	β	max iter
SimpleMF	0.812	20	0.001	0	10
TB-IP SimpleMF	0.800	20	0.001	0	10
Simple SoReg	0.807	20	0.01	0.01	10
TB-IP SoReg	**0.794**	20	0.01	0.01	20
SocSim Simple SoReg	0.813	20	0.01	0.1	10
SocSim TB-IP SoReg	0.799	20	0.01	0.01	10

From the results, it can be evidently observed that *TB-IP SoReg* approach outperformed the other experimental methodologies. It gave the best results in both the evaluating metrics used here, RMSE and MAE as mentioned in Tables 7 and 8. This confirms our claim that combining neighborhood formation with influence propagation and social regularization produces better recommendations than the traditional social recommendation approaches.

Moreover, *SocSim TB-IP SoReg* gives the next best results in our experimental analysis. Not only does it outperform the *Simple SoReg* in RMSE, but also in MAE. This verifies that inclusion of the edge-weights (defined for influence propagation) in the similarity metric in traditional social regularization affects the matrix factorization objective function such that it enhances the social recommendations. It is so because it captures the cascading effect of influence in a social network setup and combines every step into one.

As expected, the experimental results for the traditional matrix factorization for the entire Epinions dataset is inferior to the rest of the experiments. This is true for both RMSE and MAE. Although, when neighborhood generation using $EWThr$ threshold in TB-IP algorithm is included as a pre-processing step, the recommendations improve. This was also expected as we got the best results for Yelp dataset incorporating the neighborhood formation using similar *TwoD* $EWThr$ dataset.

The time taken to generate *Two* $EWThr$ Epionions dataset was around 26 minutes. Although this neighborhood generation is an additional overhead, it produces better recommendations in every scenario. Tables 9 and 10 show the time taken for neighborhood formation as well as running the recommendation algorithm that resulted in least RMSE and MAE respectively. The time recorded is only for the most optimal parameters for each experimental setup discussed in this section.

Table 9
Time Analysis for Least RMSE

Experiment	Neighborhood (min)	Recommendations (min)	Total
SimpleMF	0	9	9
TB-IP SimpleMF	26	4	30
Simple SoReg	0	53	53
TB-IP SoReg	26	19	45
SocSim Simple SoReg	0	10	10
SocSim TB-IP SoReg	26	13	39

Table 10
Time Analysis for Least MAE

Experiment	Neighborhood (min)	Recommendations (min)	Total
SimpleMF	0	9	9
TB-IP SimpleMF	26	4	30
Simple SoReg	0	28	28
TB-IP SoReg	26	19	45
SocSim Simple SoReg	0	10	10
SocSim TB-IP SoReg	26	7	33

7 FUTURE WORK

In future work, we plan to improve the social regularization-based recommendation methodology proposed in this work. We plan to use deep learning methodologies to effectively initialize the user and item latent vectors used in matrix factorization. This idea is utilized by Deng et al. [38] in their deep learning based matrix factorization by using deep autoencoder. Analogous to this work, they also introduce a trust regularization term and combine deep learning approach and trust regularization methodology in one framework.

We also plan to implement other approaches for combining influence propagation in the social regularization term. In this work, we simply consider mean of the correlation coefficient between two users and the edge weights obtained from the TB-IP algorithm for *SocSim*, there can be other methodologies to evaluate a weighted effect of influence propagation factor in the social regularization term.

Moreover, we also intend to use the $Katz\ Similarity$ metric to generate neighborhood and also evaluate the influence propagation algorithm for producing social recommendations.

8 CONCLUSIONS

In this research, the problem of generating social recommendations is explored in the context of influence propagation. Here, influence propagation is considered a prominent characteristic for information diffusion in a social network. We introduced a threshold-bounded influence propagation algorithm to determine this cascading effect. We established three conditions for determining the threshold for a node to get influenced. Along with this, three approaches are employed for the initial ranking of nodes. These variations are then extensively evaluated by experimental analysis on real-world datasets. The results show that node-dependent threshold conditions are a better choice than global threshold conditions for influence propagation. We subsequently used the proposed algorithm to generate social graph-based neighborhoods. These were used as input to the recommendation algorithm. This recommendation algorithm incorporates influence propagation effect to generate recommendations for similar users. Our experiments against non-socially enhanced baselines as well as traditional recommendation baselines verified our intuition that social recommender systems with influence propagation are indeed more accurate than the traditional social recommenders and conventional rating-based ones.

Literature Cited

[1] M. Granovetter, *Getting a job: A study of contacts and careers*. University of Chicago press, 2018.

[2] M. S. Granovetter, "The strength of weak ties," in *Social networks*, pp. 347–367, Elsevier, 1977.

[3] C. Aggarwal, *Recommender Systems: The Textbook*. Springer Publishing Company, Incorporated, 1st ed., 2016.

[4] I. Guy, *Social Recommender Systems*, pp. 511–543. Boston, MA: Springer US, 2015.

[5] M. Eirinaki, N. Moniz, and K. Potika, "Threshold-bounded influence dominating sets for recommendations in social networks," in *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, pp. 408–415, Oct 2016.

[6] S. Lei, S. Maniu, L. Mo, R. Cheng, and P. Senellart, "Online influence maximization," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 645–654, ACM, 2015.

[7] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, and J. Tang, "Online topic-aware influence maximization," *Proc. VLDB Endow.*, vol. 8, pp. 666–677, Feb. 2015.

[8] J. Weng, E.-P. Lim, J. Jiang, and Q. He, "Twitterrank: Finding topic-sensitive influential twitterers," in *Proc. of WSDM'10*, pp. 261–270, 2010.

[9] D. A. Kim, A. R. Hwong, D. Stafford, D. A. Hughes, A. J. O'Malley, J. H. Fowler, and N. A. Christakis, "Social network targeting to maximise population behaviour change: a cluster randomised controlled trial," *The Lancet*, vol. 386, pp. 145–153, 5 2015.

[10] M.-E. G. Rossi, F. D. Malliaros, and M. Vazirgiannis, "Spread it good, spread it fast: Identification of influential nodes in social networks," in *Proc. of WWW'15 Companion*, pp. 101–102, ACM, 2015.

[11] R. Huckfeldt and J. Sprague, "Networks in context: The social flow of political information," *American Political Science Review*, vol. 81, pp. 1197–1216, 12 1987.

[12] E. Dubois and D. Gaffney, "The multiple facets of influence: Identifying political influentials and opinion leaders on twitter," *American Behavioral Scientist*, vol. 58, no. 10, pp. 1260–1277, 2014.

[13] Y. Yang, X. Mao, J. Pei, and X. He, "Continuous influence maximization: What discounts should we offer to social network users?," in *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, (New York, NY, USA), pp. 727–741, ACM, 2016.

[14] M. Kaple, K. Kulkarni, and K. Potika, "Viral marketing for smart cities: Influencers in social network communities," in *9th IEEE International Workshop on Big Data Appications in Smart City Development*, 2017.

[15] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 57–66, ACM, 2001.

[16] A. S. Badashian and E. Stroulia, "Measuring user influence in github: The million follower fallacy," in *2016 IEEE/ACM 3rd International Workshop on CrowdSourcing in Software Engineering (CSI-SE)*, pp. 15–21, May 2016.

[17] Q. Jiang, G. Song, G. Cong, Y. Wang, W. Si, and K. Xie, "Simulated annealing based influence maximization in social networks.," in *AAAI*, vol. 11, pp. 127–132, 2011.

[18] Y. Zeng, X. Chen, G. Cong, S. Qin, J. Tang, and Y. Xiang, "Maximizing influence under influence loss constraint in social networks," *Expert Systems with Applications*, vol. 55, no. Supplement C, pp. 255 – 267, 2016.

[19] M. Gong, J. Yan, B. Shen, L. Ma, and Q. Cai, "Influence maximization in social networks based on discrete particle swarm optimization," *Information Sciences*, vol. 367-368, no. Supplement C, pp. 600 – 614, 2016.

[20] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on*, pp. 39–43, Oct 1995.

[21] D. Kempe, J. Kleinberg, and Éva Tardos, "Maximizing the spread of influence through a social network," *Theory of Computing*, vol. 11, no. 4, pp. 105–147, 2015.

[22] S. Hangal, D. Maclean, M. S. Lam, and J. Heer, "All friends are not equal: Using weights in social graphs to improve search."

[23] J. M. C. Miller McPherson, Lynn Smith-Lovin, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, pp. 415–444, 2001.

[24] J. Tang, X. Hu, and H. Liu, "Social recommendation: a review.," *Social Netw. Analys. Mining*, vol. 3, no. 4, pp. 1113–1133, 2013.

[25] I. Guy and D. Carmel, "Social recommender systems," in *Proc. of WWW'11*, pp. 283–284, ACM, 2011.

[26] J. Tang, H. Gao, X. Hu, and H. Liu, "Exploiting homophily effect for trust prediction," in *Proc. of WSDM '13*, pp. 53–62, ACM, 2013.

[27] H. Ma, I. King, and M. R. Lyu, "Learning to recommend with social trust ensemble," in *Proc. of SIGIR '09*, pp. 203–210, ACM, 2009.

[28] J. Tang, H. Gao, and H. Liu, "mtrust: Discerning multi-faceted trust in a connected world," in *Proc. of WSDM '12*, pp. 93–102, ACM, 2012.

[29] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. of RecSys '10*, pp. 135–142, ACM, 2010.

[30] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 287–296, ACM, 2011.

[31] P. Massa and P. Avesani, "Controversial users demand local trust metrics: An experimental study on epinions.com community," in *Proc. of AAAI-05*, 2005.

[32] J. Golbeck, "Trust on the world wide web: a survey," *Found. Trends Web Sci.*, vol. 1, no. 2, pp. 131–197, 2006.

[33] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of the 2007 ACM conference on Recommender systems*, pp. 17–24, ACM, 2007.

[34] I. Guy, M. Jacovi, E. Shahar, N. Meshulam, V. Soroka, and S. Farrell, "Harvesting with sonar: The value of aggregating social network information," in *Proc. of CHI '08*, pp. 1017–1026, ACM, 2008.

[35] P. Victor, M. De Cock, and C. Cornelis, "Trust and recommendations," in *Handbook of Recommender Systems*, pp. 645–675, springer, 2010.

[36] I. Varlamis, M. Eirinaki, and M. Louta, "A study on social network metrics and their application in trust networks," in *Proc. of ASONAM 2010*, pp. 168–175, IEEE, 2010.

[37] I. Varlamis, M. Eirinaki, and M. Louta, "Application of social network metrics to a trust-aware collaborative model for generating personalized user recommendations," in *The Influence of Technology on Social Network Analysis and Mining*, pp. 49–74, Springer Vienna, 2013.

[38] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu, "On deep learning for trust-aware recommendations in social networks," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 5, pp. 1164–1177, 2017.

[39] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[40] Q. Liu, B. Xiang, N. J. Yuan, E. Chen, H. Xiong, Y. Zheng, and Y. Yang, "An influence propagation view of pagerank," *ACM Trans. Knowl. Discov. Data*, vol. 11, pp. 30:1–30:30, Mar. 2017.

[41] J. D. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proceedings of the 22nd international conference on Machine learning*, pp. 713–719, ACM, 2005.

[42] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pp. 43–52, Morgan Kaufmann Publishers Inc., 1998.

[43] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *International semantic Web conference*, pp. 351–368, Springer, 2003.

[44] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 2, 2007.

Appendix A

In this section, we have included detailed results of the experimental analysis mentioned in Section 6.3. We have discussed results of all the experiments sequentially.

Table 11 shows the time taken to form neighborhood in each of the experimental setup. Moreover, it also states the time taken to perform 5-fold cross-validation (cv) to generate recommendations in the respective experiments. As expected, the total time taken to run *TB-IP SimpleMF* and *SimpleMF* experiments is the least. Additionally, *SocSim TB-IP SoReg* and *TB-IP SoReg* takes very less time to run when social regularization is included in matrix factorization. This is so because, the neighborhood that we had generated using TB-IP algorithm had less number of users in comparison to the experimental setup in which the entire Epinions dataset is considered.

Table 11
Detailed Time Analysis for 5-Fold Cross Validation

Experiment	**Neighborhood (min)**	**Recommendations (min)**	**Total**
SimpleMF	0	149	149
TB-IP SimpleMF	26	106	132
Simple SoReg	0	455	455
TB-IP SoReg	26	359	385
SocSim Simple SoReg	0	391	391
SocSim TB-IP SoReg	26	303	329

In Tables 12 and 13, RMSE and MAE of all the possible combination of parameters used in *SimpleMF* setup are mentioned. The best results are highlighted in the tables. We have followed the same acronyms for the tables here as before.

Table 12
RMSE for SimpleMF Experimental Setup

max iter	**f \ λ**	**0.1**	**0.01**	**0.001**
10	10	1.076964205	1.076896083	1.076934952
	20	1.062708070	1.062659148	**1.062620769**
20	10	1.114222815	1.114315574	1.114266456
	20	1.124565108	1.124677841	1.124649342

Table 13
MAE for SimpleMF Experimental Setup

max iter	**f \ λ**	**0.1**	**0.01**	**0.001**
10	10	0.829608288	0.829682485	0.829547740
	20	0.812410723	0.812298487	**0.812229840**
20	10	0.838008626	0.838109076	0.837997270
	20	0.844606051	0.844749035	0.844703202

Tables 14 and 15 show the RMSE and MAE values of all the possible combination of parameters used in *TB-IP SimpleMF* setup respectively. Tables 16 and 17 show the RMSE and MAE values of all the possible combination of parameters used in *Simple SoReg* setup respectively.

Table 14
RMSE for TB-IP SimpleMF Experimental Setup

max iter	**f \ λ**	**0.1**	**0.01**	**0.001**
10	10	1.052539859	1.052451908	1.052504903
	20	1.062267102	1.046333444	1**.046311493**
20	10	1.093338474	1.093105663	1.093126753
	20	1.103100668	1.103168478	1.103132517

Table 15
MAE for TB-IP SimpleMF Experimental Setup

max iter	**f \ λ**	**0.1**	**0.01**	**0.001**
10	10	0.807431408	0.807297063	0.807205498
	20	0.812276228	**0.800284169**	0.800128228
20	10	0.823400967	0.823512862	0.823309829
	20	0.830967050	0.831204268	0.830973575

Table 16
RMSE for Simple SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	1.120000000	1.063100000	1.071423800
		20	1.107985217	1.054069785	1.085805614
	20	10	1.140790123	1.068355540	1.074502559
		20	1.099449052	**1.045999900**	1.047752697
0.01	10	10	1.167500000	1.110000000	1.106218600
		20	1.095056623	1.048117149	1.053495749
	20	10	1.108674082	1.061993438	1.070086189
		20	1.096531629	1.048151569	1.054195080

Table 17
MAE for Simple SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	0.967000000	0.880000000	0.871530000
		20	0.914941000	0.822763182	0.822842118
	20	10	0.940615973	0.836382831	0.835919803
		20	0.904655293	0.811252440	0.806546247
0.01	10	10	0.915480000	0.821920000	0.826070000
		20	0.894397308	**0.807450625**	0.808287380
	20	10	0.897854001	0.819799057	0.823588246
		20	0.889326358	0.806705434	0.807775017

Tables 18 and 19 show the RMSE and MAE values of all the possible combination of parameters used in *TB-IP SoReg* setup respectively. Again, Tables 20 and 21 show the RMSE and MAE values of all the possible combination of parameters used in *SocSim Simple SoReg* setup respectively. Lastly, Tables 22 and 23 show the RMSE and MAE values of all the possible combination of parameters used in *SocSim TB-IP SoReg* setup respectively.

Table 18
RMSE for TB-IP SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	1.141165121	1.055955935	1.053855079
		20	1.089382883	1.038366330	1.065915803
	20	10	1.119788790	1.037713423	1.042248132
		20	1.078597795	**1.030497489**	1.031124000
0.01	10	10	1.102501521	1.042234952	1.046596287
		20	1.075872078	1.032198057	1.037119556
	20	10	1.089848405	1.042075753	1.046098954
		20	1.077922764	1.032286359	1.037220516

Table 19
MAE for TB-IP SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	0.939489341	0.825531887	0.818279916
		20	0.893455645	0.808290282	0.809571841
	20	10	0.918035730	0.804803435	0.803131349
		20	0.881428158	0.797944774	0.793113321
0.01	10	10	0.896230157	0.803078042	0.803402346
		20	0.873286954	0.794767805	0.795449893
	20	10	0.878709931	0.801494306	0.801868999
		20	0.868801273	**0.794147338**	0.794889564

Table 20
RMSE for SocSim Simple SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	1.120559564	1.070147797	1.085467352
		20	1.096650740	**1.060717589**	1.085431016
	20	10	1.113503512	1.117381664	1.154154447
		20	1.104190341	1.107923501	1.158282022
0.01	10	10	1.120750522	1.070163595	1.085376777
		20	1.095933605	1.060851330	1.085293043
	20	10	1.113402114	1.117384799	1.155583409
		20	1.104006041	1.107805605	1.158396523

Table 21
MAE for SocSim Simple SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	0.915509056	0.822283838	0.825507413
		20	0.895267610	0.812901919	0.822543741
	20	10	0.900212868	0.844402659	0.860890173
		20	0.893608563	0.840255740	0.866145151
0.01	10	10	0.915803441	0.822498000	0.825770953
		20	0.894607051	**0.813273497**	0.822696907
	20	10	0.899973296	0.844487871	0.861782188
		20	0.893296766	0.839896598	0.866440649

Table 22
RMSE for SocSim TB-IP SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	1.102588583	1.050350556	1.062870468
		20	1.077028504	**1.043075758**	1.065970332
	20	10	1.095874171	1.094906520	1.128007157
		20	1.083935785	1.084471390	1.129180180
0.01	10	10	1.102566624	1.050713291	1.063209012
		20	1.076270974	1.043139051	1.065835501
	20	10	1.095298643	1.094464795	1.127481043
		20	1.084347501	1.084212342	1.129424158

Table 23
MAE for SocSim TB-IP SoReg Experimental Setup

β	**max iter**	**f** $\setminus$ λ	**0.1**	**0.01**	**0.001**
0.1	10	10	0.895663353	0.805528455	0.807901500
		20	0.874097645	0.800148206	0.809403199
	20	10	0.880341525	0.828503928	0.844229150
		20	0.871215303	0.823678030	0.847012599
0.01	10	10	0.895728180	0.806085945	0.808308126
		20	0.873297756	**0.799538188**	0.809300084
	20	10	0.879881653	0.828267674	0.843404193
		20	0.871628896	0.823242614	0.847375519